an object inherently has "attributes". This terminology is expressly noted in the preamble of Claim 1 as amended to address the antecedent objection made by the Examiner, in addition to other clarifying amendments noted below.

IV. Rejections under 35 U.S.C. §102(e)

A. The Faustini Patent

The Examiner has cited the Faustini Patent (U.S. Pat. No. 5,842,020) under 35 U.S.C. §102(e) in rejecting Claim 1. The Faustini Patent describes a system in which objects of an object-oriented application may be defined to have an object editor as an integral part of each object within the application. For example, at column 153, line 58 to column 154, line 2, the Faustini Patent states:

> "Dynamic editing is accomplished by providing each component that would have need of an editor with that capability as an integral part of the class template from which it is instantiated."

Hence, in the system described in the Faustini patent, the properties of various objects are made editable only because the objects themselves have been coded to include a self-editing function. The complete context of column 154 containing the portion thereof applied by the Examiner relative to the Applicant's recited inspector, discussed below, states:

> "Thus, when such edit capable components are instantiated in either the logical view 402 or the physical view 500, their built-in customizer or edit widow 1104 is invoked, see FIG. 11, and opens automatically. The editor appears in the view ready for use to change or customize the properties of the component, in this case scrollbar 604b, based on user interaction with the customizer or editing window 1104."

The Faustini patent here refers clearly to "edit capable components" as those

components instantiated from a class template that includes a "built-in customizer" as an integral part thereof. Such built-in editing capability is in accordance with the stated objective of the Faustini patent at column 5, lines 23-33, which refers to:

> "[...] defining an editor window in predetermined class templates as a method corresponding to the editor. Then, when a component is instantiated from one of said predetermined classes, the editor is automatically opened to permit the user to make changes in the component's properties."

Thus, the central teaching of the Faustini patent is to define, for each object desired to be edit-capable, an editor within the object itself, so that the editor is invoked wherever such object is deployed. The editable objects described in the Faustini patent have development and configuration capabilities that are pre-programmed into them. Therefore, the modification capabilities are limited to the objects and features that were anticipated during the creation of the application. Additionally, such editable objects would necessarily remain modifiable by users when deployed, which is not desirable for applications that are intended to remain static, or at least subject to consistent management control, in their deployment environments.

B. Applicant's Invention

The present Application describes a system by which object-oriented applications can be modified within an execution environment, when that execution environment is provided with access to a set of "inspector documents" which define inspector objects capable of making changes to the attributes of application objects. An important distinction between the system described in the Application and that of the Faustini patent is that objects of the present invention are not required to be pre-programmed to include "edit" functions. Rather, separate inspector objects can be defined for generic attributes of objects, and for specific attributes of objects, and these inspectors can be accessed and activated by a document server that maintains an inventory of deployed objects in the execution environment. It should be

appreciated that the application itself can be then modified via the inspectors while it is deployed, then archived, and then later re-deployed in its modified form in an execution environment which does not have access to the inspector document server. Hence the modified application can be deployed in an execution environment where user modification of the application is not desired, and the application does not carry its own editing methods with it. However, the system described in the present application allows an execution environment to be augmented by providing access to the inspector objects, to permit live modification of the application in the execution environment itself, rather than in a separate development environment. The Applicant's claims recite features of the disclosed system that provide the ability to permit such live modification of a deployed application.

Claim 1 is hereby amended to clarify the nature of the relationship of the recited inspector method and the recited application object upon which the inspector method operates to permit modification. As noted in Section A above, the "edit window" of the Faustini patent relied upon by the Examiner as corresponding to Applicant's recited "inspector method" is a built-in function of the object desired to be edited. Claim 1 is hereby amended to more clearly recite the "inspector object" as being configured to communicate with the application object, and to permit modification of the object, but as a separate entity operative within the object-oriented application system instead of as a component of the application object deployed in the execution environment.

As can be seen in FIG. 2 of the Application, an application object 30 is deployed in an execution environment by de-archiver 36, which deploys the application object 30 on the basis of its definition within an object document 34. The inspector 38 corresponding to the application object 30 is deployed in the execution environment by document server 40 on the basis of an inspector document 42. In the absence of the inspector 38, the application object 30 would remain deployed and executable in the environment, but would not be modifiable, in contrast to the "editable objects" described in the Faustini patent to have their own editors built-in. Claim 1, as hereby amended, recites deployment, in the object-oriented system, of an inspector

that is configured to operate upon and to communicate with, application objects deployed in the execution environment.

The Applicant's invention, as recited in amended Claim 1, permits modification of an application object within the execution environment. Changes to an object can thus be made "on the fly" while the application executes, in contrast to the system described in the Faustini patent wherein execution is halted when an edit window is opened upon instantiation of an editable object.

For the foregoing reasons, Claim 1 as amended hereby is believed to be distinguished over the art cited by the Examiner. Claims 2 and 3, dependent thereon are also believed to be allowable for at least the same reasons. Clarifying amendments have been made to Claims 2 and 3 in view of the amendments made to claim 1.

V. New Claims

Further Claims 4 - 10 are added hereby to expressly recite further distinct features of the invention. Claims 4 - 6 are dependent on Claim 1, and are believed to be allowable for the same reasons discussed above in connection with Claim 1.

Claim 7 recites a software modification system in which a document server is provided with access to an inventory of runtime objects deployed in an execution environment, and where the document server further provides selective access to at least one inspector configured for modification of one of the deployed runtime objects. Again, such an arrangement is distinct from the system described in the Faustini patent of record, in which objects are pre-programmed to contain their own editors.

Claims 8 - 10 are dependent on Claim 7, and are supported in the Specification as follows. Claim 8 recites that the inspector is configured to generate a display of modifiable attributes of its corresponding runtime object. Such a feature is shown and described in connection with FIG. 3 of the Application. Claim 9 recites the feature

discussed at page 9, lines 18-21 wherein a modified object is subsequently re-archived in order to retain the desired modification when the object is deployed in the future. Claim 10 recites that the execution environment may comprise a web browser, as discussed in the Specification at page 18, lines 6-9.

No new matter is added hereby.

VI. Conclusion

In accordance with the foregoing amendment and remarks, the rejected claims are believed to be distinguished over the prior art of record, and in condition for allowance. Additionally, new claims are added hereby which further distinguish the present invention over the art of record.

Reconsideration of the Application as hereby amended is respectfully requested. If the Examiner believes that further discussion may lead to prompt and favorable action on the Application, the Examiner is encouraged to contact the undersigned attorney by telephone.

Respectfully submitted,

DANN, DORFMAN, HERRELL & SKILLMAN
A Professional Corporation
Attorneys for Applicant(s)

By _____
John B. Berryhill, Ph.D.
PTO Registration No. 36,452

Telephone: (215) 563-4100
Facsimile: (215) 563-4044

ATTACHMENT A

1. An object oriented software application system, having an execution environment in which an object-oriented application is deployed as a collection of application objects, each application object having attributes defining properties thereof, the system comprising:

an inspector [method associated with] configured to permit modification of at least one application object in [an] the object-oriented application, for communicating information pertaining to the attributes of the application object while the application object is deployed in [an] the execution environment; and

a document server for maintaining an inventory of objects deployed in the execution environment, the document server providing a user interface to the inspector method [of] for communicating with said one application object.

2. The system of claim 1 wherein the document server is configured to actuate the [inspector method of the object,] inspector, and wherein the inspector [method] is configured to generate a display of attributes defining the object.

3. The system of claim 2 wherein the inspector [method] is [configure] configured to allow user modification of the attributes shown in the display and to apply such modification to the attributes of the application object in the memory of [the computer.] the execution environment.

4. The system of claim 1 wherein the inspector is configured to permit user modification of the application object during execution of the object-oriented application.

5. The system of claim 1 wherein the document server is configured to maintain an inventory of objects deployed in the execution environment, and to retrieve and instantiate a corresponding inspector for a deployed object desired to be modified.

6. The system of claim 1 further comprising a library of objects accessible to the document server, for permitting addition of objects from the library to the object-oriented application.

7. An object oriented software modification system for use in an execution environment, comprising:

at least one inspector, adapted to communicate with at least one runtime object in the execution environment, for communicating and altering attribute information associated with the runtime object while the runtime object is deployed in the execution environment; and

a document server for accessing an inventory of runtime objects deployed in the execution environment, and configured for selecting said one inspector for deployment in the execution environment, whereby attributes of said runtime object can be selectively modified by operation of the inspector while the runtime object is deployed in the execution environment.

8. The system of claim 7, wherein the inspector is configured to generate a display of modifiable attributes of the runtime object.

9. The system of claim 7, further comprising means for archiving the runtime object after it has been modified.

10. The system of claim 7 wherein the execution environment comprises a web browser.

ATTACHMENT B

1. An object oriented software application system, having an execution environment in which an object-oriented application is deployed as a collection of application objects, each application object having attributes defining properties thereof, the system comprising:

an inspector configured to permit modification of at least one application object in the object-oriented application, for communicating information pertaining to the attributes of the application object while the application object is deployed in the execution environment; and

a document server for maintaining an inventory of objects deployed in the execution environment, the document server providing a user interface to the inspector method for communicating with said one application object.

2. The system of claim 1 wherein the document server is configured to actuate the inspector, and wherein the inspector is configured to generate a display of attributes defining the object.

3. The system of claim 2 wherein the inspector is configured to allow user modification of the attributes shown in the display and to apply such modification to the attributes of the application object in the memory of the execution environment.

4. The system of claim 1 wherein the inspector is configured to permit user modification of the application object during execution of the object-oriented application.

5. The system of claim 1 wherein the document server is configured to maintain an inventory of objects deployed in the execution environment, and to retrieve and instantiate a corresponding inspector for a deployed object desired to be modified.

6. The system of claim 1 further comprising a library of objects accessible to the document server, for permitting addition of objects from the library to the object-oriented application.

7. An object oriented software modification system for use in an execution environment, comprising:

at least one inspector, adapted to communicate with at least one runtime object in the execution environment, for communicating and altering attribute information associated with the runtime object while the runtime object is deployed in the execution environment; and

a document server for accessing an inventory of runtime objects deployed in the execution environment, and configured for selecting said one inspector for deployment in the execution environment, whereby attributes of said runtime object can be selectively modified by operation of the inspector while the runtime object is deployed in the execution environment.

8. The system of claim 7, wherein the inspector is configured to generate a display of modifiable attributes of the runtime object.

9. The system of claim 7, further comprising means for archiving the runtime object after it has been modified.

10. The system of claim 7 wherein the execution environment comprises a web browser.